Analysing and Demonstrating Tool-Supported Customizable Task Notations

CELIA MARTINIE, ICS-IRIT, Université Paul Sabatier, Toulouse 3, France
PHILIPPE PALANQUE, ICS-IRIT, Université Paul Sabatier, Toulouse 3, France & Technical University Eindhoven, Department of Industrial Design, Eindhoven, Netherlands
ELODIE BOUZEKRI, ICS-IRIT, Université Paul Sabatier, Toulouse 3, France
ANDY COCKBURN, University of Canterbury, Christchurch, New Zealand
ALEXANDRE CANNY, ICS-IRIT, Université Paul Sabatier, Toulouse 3, France
ERIC BARBONI, ICS-IRIT, Université Paul Sabatier, Toulouse 3, France

When task descriptions are precise they can be analysed to yield a variety of insights about interaction, such as the quantity of actions performed, the amount of information that must be perceived, and the cognitive workload involved. Task modelling notations and associated tools provide support for precise task description, but they generally provide a fixed set of constructs, which can limit their ability to model new and evolving application domains and technologies. This article describes challenges involved in using fixed notations for describing tasks. We use examples of recognized tasks analysis processes and their phases to show the need for customization of task notations, and through a series of illustrative examples, we demonstrate the benefits using our extensible task notation and tool (HAMSTERS-XL).

KEYWORDS

Task Modelling, Model-Based Design of Interactive Systems, Notations, Tools

1 INTRODUCTION

Task Analysis is a cornerstone of User Centered Design approaches, aiming to collect information from users about the work they are doing and the way they perform it. According to Johnson [24] "any Task Analysis is comprised of three major activities; first, the collection of data; second, the analysis of that data; and third, the modelling of the task domain" (p.165). The means for representing the outcomes of task analysis has important implications for the value and insight gained from the process, not least because any omissions cannot be discussed (among the stakeholders) or taken into consideration in later design phases.

The expressive power of the notation used to store and organize the information collected is thus a key element that is put forward by researchers proposing new notations [6]. Since the seminal HTA notation proposed by [1][2], relatively few notations to describe user tasks have been proposed and they all tend to remain unchanged after their creation. While the notations

https://doi.org/10.1145/3331154

remain largely constant, their associated tools typically evolve to address new challenges. For instance, Paterno's team has proposed several tools exploiting CTT notation since its creation in 1997 [55]: the original CTTe tool [42] supported editing, simulating and verifying CTT task models; CTTEVis [57] added support for visualization; and a new tool added support for collaborative modelling [31]. The stability of notations contrasts with the constant evolution of application domains, technologies, and the nature of work operators' work. This evolution can create a gap between what the notation can describe and the actual work, limiting the scope and potential benefits of using the notation at all. For instance, a notation like KMAD [6] or CTT [55] would produce the same representation of tasks for interacting with a calculator regardless of whether the calculator was a physical device, a desktop application, or an app on a mobile phone. This lack of precision and detail make it impossible for analysts to assess the interaction implications of moving from one technology to the other one.

Designing a notation and its associated tools is a complex and time-consuming process. It is therefore unsurprising that evolutions are limited and can only be produced by research teams with long lasting interests and recurrent funding. This is part of the reason for why tasks notations have been built to address a broad spectrum of users' work, covering standard application domains and addressing widely available interaction techniques (such as WIMP interfaces). However, extending notations and tools to support task analysis in specialized domains requires either new notations or extensible/customizable notations. The need of extensions is however well perceived and, for instance, temporal operators of CTT notation have been extended in [61] to be able to describe user interface behaviour with task models. This is a particular extension as the objective was to use the extended notation for a purpose different than describing users' tasks. A similar approach has been followed in [32] where preconditions were added to elementary tasks with the objective of generating User Interfaces. However, these extensions were then used for describing preconditions in task models themselves [16]. This demonstrates the need and the usefulness of customizing notations to increase their suitability for modelling.

This article addresses customization of task analysis at two levels: first, a notation for modelling tasks is modified with features that enable the notation to be customized, allowing new interactive elements to be described using the notation; second, an associated modelling tool is extended to support the editing and simulation of customized task models. Although our description and demonstration of these new abilities is based on an existing task modelling notation, the concepts are generic enough to be embedded in other platforms.

The article is structured as follows: section 2 presents the motivation underlying the customization of tasks descriptions. It first highlights the importance of task analysis in UCD approaches. Then, using an illustrative example, it demonstrates the increasing gap between the expressive power of task modelling notations and the actual work context and technologies of interaction. Section 3 presents related work on task modelling techniques and how these techniques can be embedded in the generic process of task analysis. This generic process makes explicit the expected outcome of the task analysis and how these outcomes can be supported by modelling tasks and analysing those task models. Section 4 describes issues associated with the selection and customization of task modelling techniques, and the customization process is exemplified with the HAMSTERS task modelling notation [40][35]. Section 5 presents the HAMSTERS-XLE environment, a computer-aided software environment that supports task modelling using the customizable HAMSTERS-XL notation. Section 6 further demonstrates the value of the tools and methods through a set of examples of customized task modelling.

2 MOTIVATIONS

This section highlights the current limitations of task modelling notations when used for non-standard application domains and technologies. Starting with a list of potential benefits for

exploiting task analysis in User Centered approaches, we demonstrate that static notations constrain the benefits of notation-based task analysis.

2.1 Scope and objectives of task analysis in UCD approaches

Task analysis is a fundamental technique in Human-Computer Interaction [19], used to understand the users' goals and tasks as well as their means for completing them. Many instances of this technique exist to provide support for the design and evaluation of interactive systems. The following non-exhaustive list identifies some of the objectives of task analysis, highlighting its broad range of uses:

- Identification and description of the required functions for interactive system [19] [55],
- Identification and description of knowledge required to perform a task [10] [25] [39] [59],
- Identification and description of the temporal ordering of the user actions with the system [26] [34] [55],
- Identification and description of the different user roles and actors for groupware systems [58] [62],
- Identification and description of workflow between users for collaborative activities [58] [62],
- Understanding of an application domain [54],
- Recording the results of interdisciplinary discussions [46] [54],
- Production of scenarios for user evaluation [65] as well identification and generation of relevant test cases [8],
- Heuristic evaluation of usability of interactive applications [7] [58],
- Predictive assessment of task complexity and workload (motor, cognitive, perceptive) [45],
- Predictive assessment of user performance when interacting with the system [23],
- Exploration of the range of ways in which the system may be used [58],
- Preparation of training programs [1] [2] [38],
- Production of user manual [18] [53] and contextual help [20] [47] [53] [54],
- Identification and description of possible allocation of functions and tasks between the system and the user [35] [48],
- Designing new applications consistent with the user conceptual model [54],
- Identification and description of potential user errors [14] [61]

Task analysis is thus a pillar of UCD approaches for the design of interactive systems. If the results of task analysis do not contain sufficient information, the missing information may negatively affect the design of the interactive system and its usability. The following section illustrates this problem.

2.2 The effectiveness of task analysis depends on modelling the selected notations' expressiveness

The models that result from task analysis will differ according to the features of the selected modelling language or notation. These modelling differences are likely to illuminate (or suppress) different aspects of the interaction. It is therefore important to choose the most suitable task modelling technique, i.e. the notation with the most suitable expressiveness, which highlights the aspects that are relevant to the goals of his/her analysis.

This section demonstrates by example that the effectiveness of a task analysis process depends on the expressiveness of the notation used to describe the tasks. In this demonstration, two different task modelling notations are employed to describe the simple task "Withdraw money" from two versions of an Automated Teller Machine (ATM). The analysis focuses on the user's cognitive activities and the quantity of physical movement required. The first version of the ATM

has a keyboard and a non-tactile screen (depicted in Fig. 1a). The second version of the ATM has a fully tactile screen (depicted in Fig. 1b).

a)  b) 

Fig. 1. Example of ATM with a) hard keys and output only screen and b) fully tactile screen

In order to perform the analysis, we first gather the set of user tasks, and we then record the data using two task modelling notations: the CTT (ConcurTaskTrees) notation [55] and the HAMSTERS notation and tool [40]. The CTT notation and CTTE tool are widely used for task analysis (more than 10K registered users) [64]. HAMSTERS aims at editing and simulating user tasks with large-scale interactive systems [40], and it provides support for task-centered design and development of interactive systems [49]. Both notations provide support for describing user tasks in a hierarchical and temporally ordered way.

Fig. 2 to Fig. 7 present the CTT task models (global views and focus views) for the user task "Select amount" required to reach the user goal "Withdraw Money" with the ATM. Fig. 3 focusses on the part "a" of Fig. 2, concerning amount selection using the hard keys ATM. First, the system displays available amounts ("Display available amounts" system task with perceivable object, represented with an arc on top of the system task) that the user perceives ("Perceive available amounts" user task). The user chooses needed amount among the list of available amounts ("Choose amount" user task). Then, the user focus on the area where the needed amount is displayed.

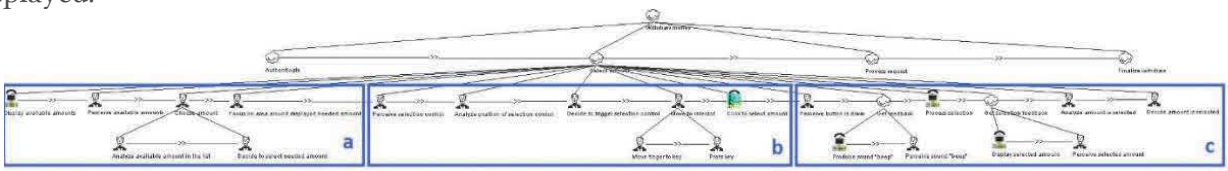

Fig. 2. CTT Task model of the user task « select amount » using "hard keys + output only screen" ATM

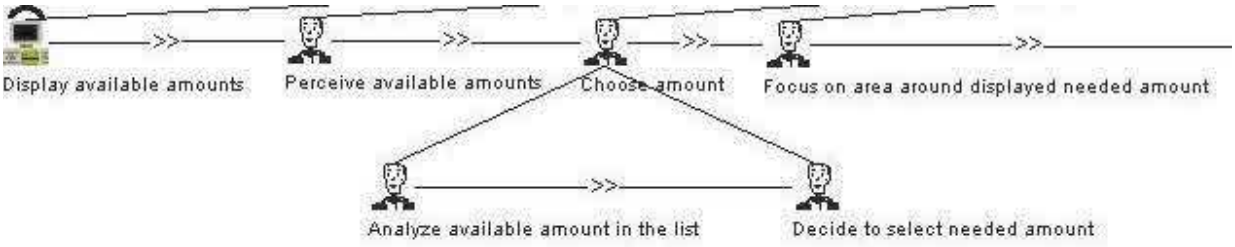

Fig. 3. Focus on part « a » of the CTT task model of the user task "Select amount" of Fig. 2

Fig. 4 presents the following tasks of the Fig. 3 and the part “b” of Fig. 2. The user perceives the selection control, analyses its position and decides to trigger it. Then, the user moves the finger on key and presses it to trigger the interactive task to select amount.

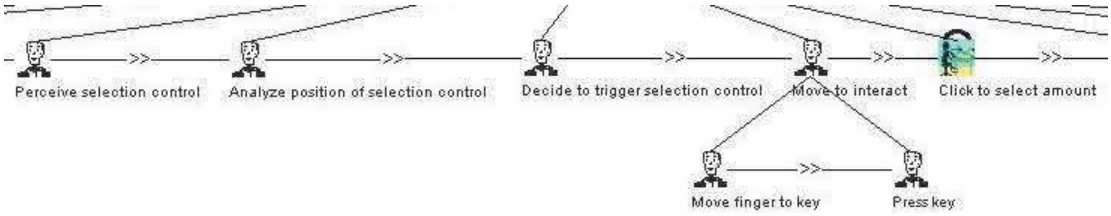

Fig. 4. Focus on part « b » of the CTT task model of the user task “Select amount” of Fig. 2

Fig. 5 describes the part “c” of Fig. 2. The user perceives that the button is down and gets a sound feedback (“Produce sound “beep”” system task with perceivable object and “Perceive sound “beep”” user task, represented with an arc on top of the system task) from the ATM. Then, the system processes the selection and gives a selection feedback (it displays selected amount). Finally, the user perceives and analyses the selected amount. Then, the user decides that the correct amount is selected.

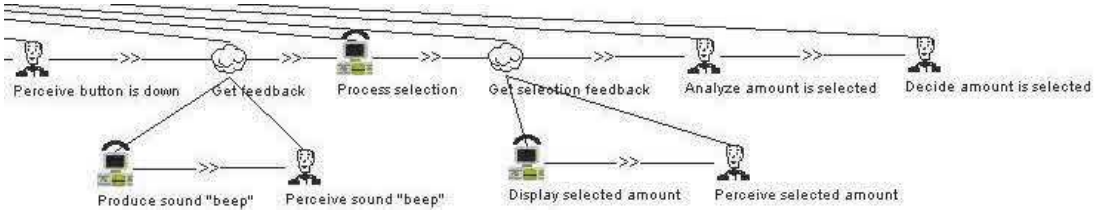

Fig. 5. Focus on part « c » of the CTT task model of the user task “Select amount” of Fig. 2

Fig. 6 proposes an overview of the user task “Select amount” required to reach the user goal “Withdraw Money” with a fully tactile screen.

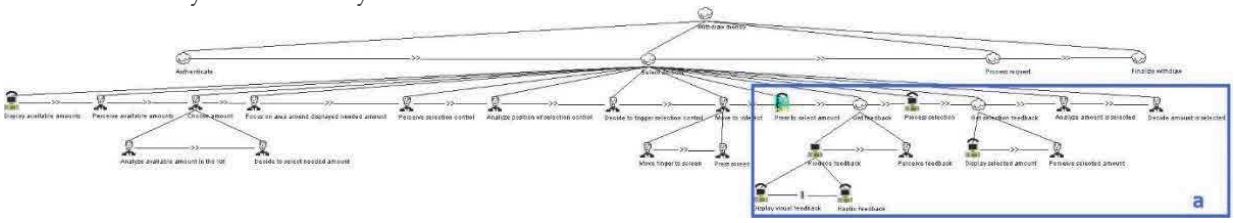

Fig. 6. CTT Task model of the user task « select amount » with the “fully tactile screen” version of the ATM

Fig. 7 focuses on the part “a” of Fig. 6 where we see a tiny difference in the first produced feedbacks. Indeed, the system produces a haptic feedback (“Haptic feedback” system task with perceivable object, represented with an arc on top of the system task) whereas the system produces a sound feedback in the hard keys version of the ATM.

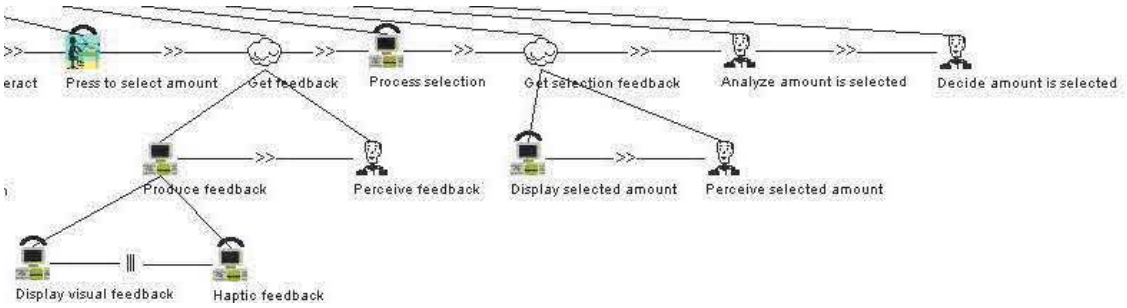

Fig. 7. Focus on part « a » of the CTT task model of the user task "Select amount" of Fig. 6

It is important to note that the task models for describing the user tasks with both types of technologies ("hard keys and output only screen" version and "fully tactile screen" version) are very similar. The structures of the models are the same, the number of tasks are the almost the same, the representation of the nature of the tasks (i.e. the types of tasks) are the same. The main differences are the text labels displayed under the tasks. Then, it is not easy to analyse the differences of usage between the two versions of the ATM.

From these task models, we extract the precise number of tasks per types and we build the table of comparison of the number of tasks for the two versions of the ATM (see Table 1).

Table 1. Comparison of the number of tasks per types for the two version of the ATM

	CTT models for "hardkeys + output only display" version of the ATM	CTT models for "fully tactile display" version of the ATM
User	14	13
Interactive	1	1
System	4	4
Objects perceivable by the user	5	6

The data from the CTT models, summarized in Table 1, suggests that the two alternative ATM designs are similar in terms of user actions. However, it is not possible to analyse whether the distribution of cognitive and motoric activities is also similar – CTT lacks the expressivity to explicitly encode these features.

We now use the HAMSTERS notation to describe the user tasks for the same two versions of the ATM. Fig. 8 and Fig. 9 presents the HAMSTERS models for the user task "Select amount" required to reach the user goal "Withdraw Money" with the ATM.". In Fig. 8, the user faces an ATM with hard keys and a non-tactile screen. Once the user is authenticated (folded abstract task "Authenticate"), the system "Display(s) available amounts" and the user decides the amount to withdraw ("Inf: Amount needed") from the displayed list ("Perceive available amount" and "Choose amount"). The user needs to "Focus on area around displayed needed amount" in order to "Perceive selection control" matching the amount needed. When the user "Decide(s) to trigger selection control", she/he "Move(s) finger to key", "Press (the) key" and "Perceive(s) button is down". This triggers the system output "Select amount" for which the system provides a sound feedback ("Produce sound beep") that is perceived by the user. The system also provide a visual selection feedback through the "Display selected amount" output task.

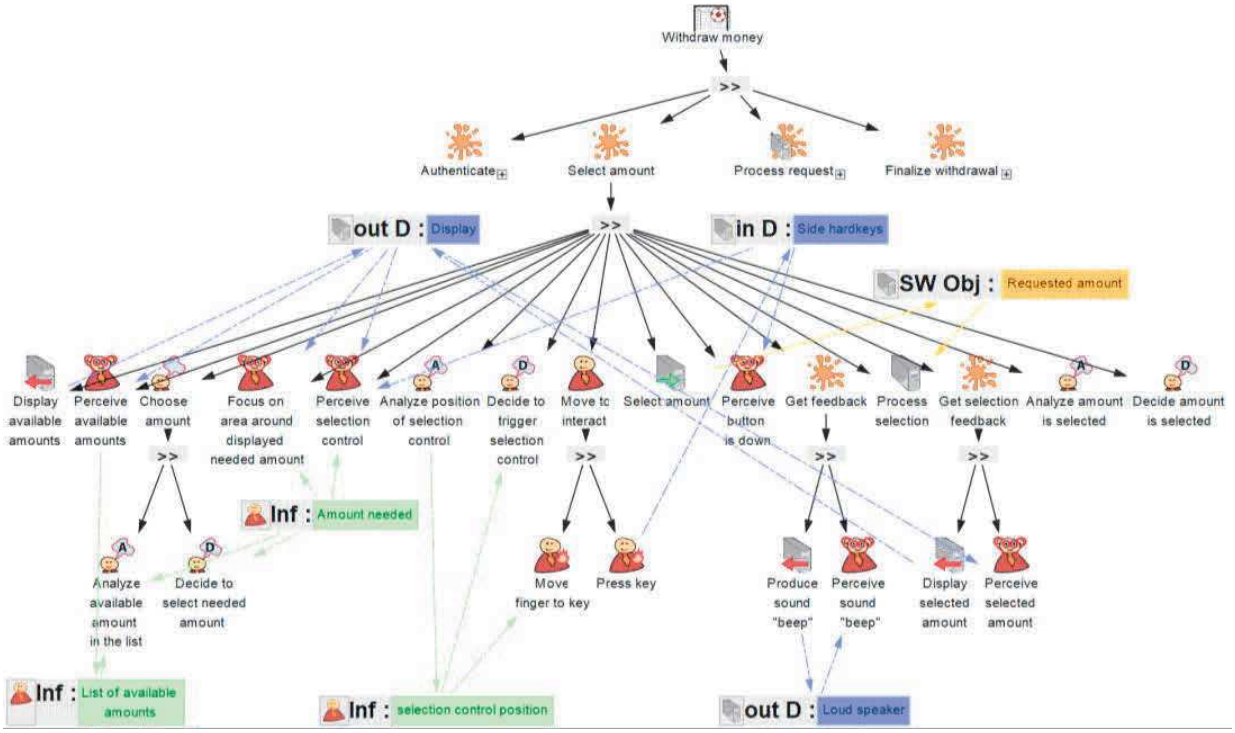

Fig. 8. HAMSTERS Task model of the user task "select amount" with the "hard keys + output only screen"

In Fig. 9, the user exploits an ATM with a fully tactile screen. Here we observe that whilst the initial system and user behaviours are the same, the feedback provided by the system is different. Indeed, the system do not produce a beep: the system "Display(s) visual feedback" and produces "Haptic feedback" that the user should perceive.

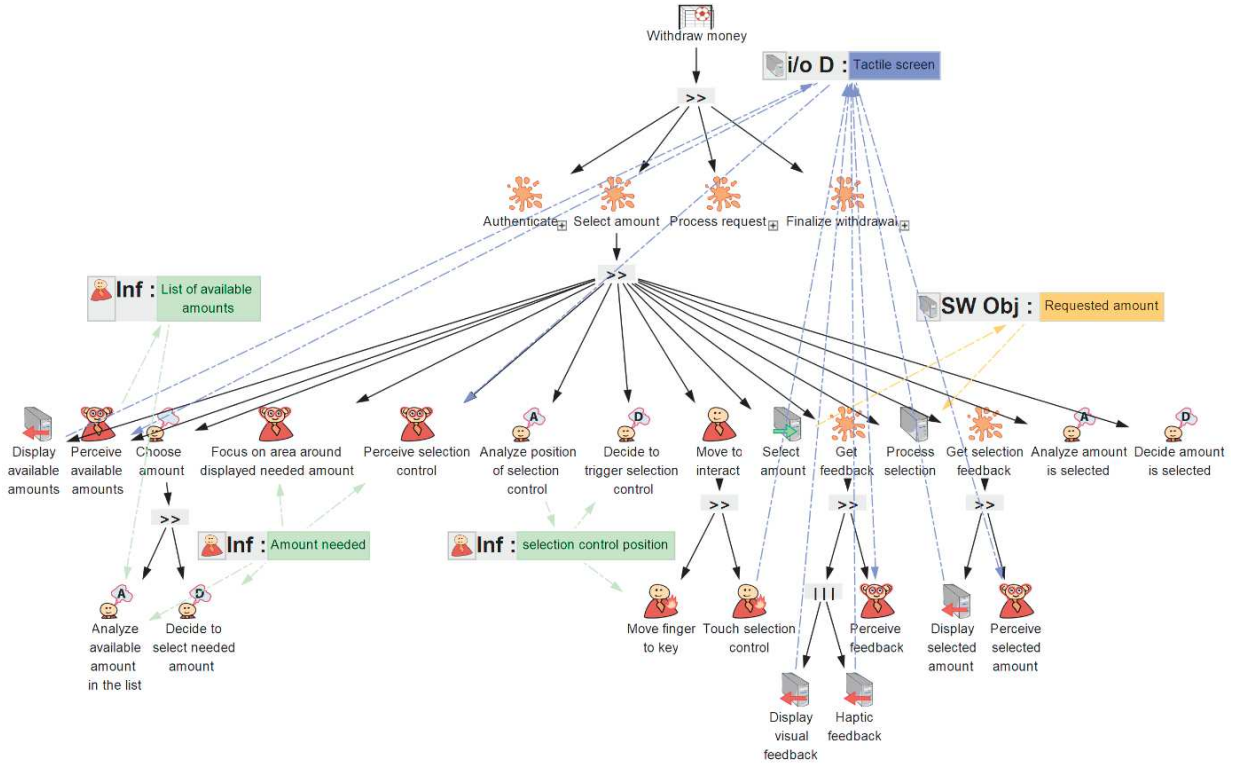

Fig. 9. HAMSTERS Task model of the user task "select amount" with the "fully tactile screen"

It is important to note that the task models for describing the user tasks with both types of technologies ("hard keys and output only screen" version and "fully tactile screen" version) are very similar. The structures of the models are the same, the number of tasks are the almost the same. The representation of the nature of the tasks (i.e. the types of tasks) is broader but there are no visible differences between the natures of the tasks for the two versions of the ATM. There are also differences in the text labels displayed under the tasks. From these task models, we extract the number of tasks per types and we update the table of comparison of the number of tasks for the two versions of ATM (see Table 2).

Table 2. Comparison of the number of tasks per types and of the types of manipulated data

	CTT models for "hardkeys + output only display" ATM	CTT models for "fully tactile display" ATM	HAMSTERS models for "hardkeys + output only display" ATM	HAMSTERS models for "fully tactile display" ATM
User	14	13	15	14
Perceptive	NA	NA	6	5
Motor	NA	NA	2	2
Cognitive	NA	NA	1	1
Cognitive analysis	NA	NA	3	3
Cognitive decision	NA	NA	3	3
Interactive	1	1	1	1
Input	NA	NA	1	1
Input/Output	NA	NA	0	0
System	4	4	4	5
Output	NA	NA	3	4
Processing	NA	NA	1	1
Manipulated data	5	6	7	5
Object	5	6	1	1
Information	NA	NA	3	3
Input device	NA	NA	1	0
Output device	NA	NA	2	0
Input/output device	NA	NA	0	1

The HAMSTERS model (right columns of Table 2) permits easy comparison of the cognitive demands and number of motor actions with the two ATM designs (they are similar). However, neither the CTT nor HAMSTERS models reveals data on the quantity of movement for the two ATM demands – neither notation provides the expressiveness required for this analysis.

This example of task analysis performed with two different task modelling notations illustrates the fact that the scope of task analysis relies on the notation used to describe user tasks. Having a notation that embeds all the required elements for the targeted analysis is thus important to reach all the goals of the targeted analysis.

2.3 Interactive systems and interaction techniques evolve faster than task modelling notations

Each task modelling technique initially targets a particular type of interactive system and sometimes a particular application domain (Table 3 shows some examples). However, the type of technology manipulated by the user has an impact on the user tasks. For example, from a user motoric action perspective, triggering a command by pressing a mouse button is different from triggering the same command by performing a gesture in the air. The previous section shows a simple example of this limitation, but the increasing variety and number of interaction techniques and interactive systems generates an important need of means for precisely refining the description of user tasks. Task modelling notations should support the addition of new types of user actions, as well new types of devices, data and knowledge that may be required during the performance of interactive tasks.

Some task modelling notations have been refined to support representation of specific data types and devices. For example, since the release of the TERESA framework [CITE], the CTT notation was extended to support description of the type of interactive system (desktop, mobile) manipulated by the user [42]. Another example is the K-MAD notation, which provided elements to describe large and structured conditions on objects for enabling the execution of a task [6]. The HAMSTERS notation was also modified to allow description of manipulated devices, information and physical objects [39]. However, these refinements are insufficient to cover the rapid evolution of the technologies and usages for interactive systems.

3 RELATED WORK

This section provides an overview of existing task analysis techniques and their main objectives.

3.1 Task analysis techniques and their associated task modelling notations

This section provides a non-exhaustive review of existing task analysis techniques, as summarized in Table 3. For each technique the table summarizes the primary scope and objective of the analysis (the main purpose for a designer/analyst to use the technique), output format for the results of the analysis, notation for the task models, the associated technologies for the interactive system under analysis, and an example of results of the task analysis. Table 3 illustrates the relationships between the scope and objectives of the task analysis and the characteristics of the associated task modelling notation. For example, the task analysis techniques and associated notation TKS was created to specify knowledge required by the user to use a desktop application. The elements of the TKS notation consequently embed constructs for representing procedural and declarative knowledge.

It is important to note that some tools supporting those notations offer some customization but this is never made explicit as a feature of the tools. A good example of that is CTTE [42] where each task element belongs to a category (abstract, user, system, interactive). This list of categories cannot be extended but it is possible to add a new task type to any category as an attribute of the task element. However, this information is not visible on the models and users are required to open the information about the task element to identify its task type. This demonstrates again the need for customization and the proposed approach presented in the paper supports explicit and systematic customization of task modelling notations.

Table 3. Examples of task analysis techniques and their main characteristics (chronological order)

Task analysis technique	Scope and objectives of the task analysis	Output format	Associated task modelling notation	Associated technologies for the interactive system under analysis	Examples of result of the analysis
HTA [1] Hierarchical Task Analysis	Identification of user tasks Preparation of training programs	Task model and tables of tasks	HTA	Steel production, chemical refining, power plant	Set of alternatives of user tasks, descriptions of procedures for training
GOMS [23]	Prediction of human performance while interacting with a user interface	Tables of predicted times for user actions	KLM, CMN-GOMS, NGOMSL, CPM-GOMS	Desktop computer	Estimated time to move text in a text edition tool, estimated time to learn a set of tasks
TKS [25]	Identification of the knowledge required to perform a task	Task models	TKS	Desktop computers	Spec. of procedural and declarative knowledge required to use a system
CTT [55]	Identification of user tasks, system tasks and interactive tasks Identification of temporal relationships between them Identification of UI objects perceivable by the user	Task models, scenarios	CTT	Desktop computers (has since been refined for mobile and web technologies)	Spec. of the UI dialog Percentage of user tasks covered by the interactive system
GTA [62]	Identification of user roles and user tasks for collaborative activities, identify usability problems	Task models, scenarios, flow diagrams	MAD	Desktop computers	Spec. of collaborative activities while using a groupware system
COMM [26]	Identification of user, system, interactive and group tasks, identification of temporal combinations between modalities	Task models	COMM	Desktop computers, command and control applications	Spec. of multi-user multimodal interactive systems
HAMSTERS [40]	Identification of user, system and interactive tasks, temporal relationships, manipulated data Allocation of functions Identification of cooperative activities	Task models, scenarios	HAMSTERS	Desktop computers, command and control applications	Spec. of user tasks, consistency between user tasks and system dialog model, automation possibilities, knowledge required to use the system
CoTAL [5]	Identification of roles, actors and cooperative activities	Task models	CoTAL	Desktop computers, smart rooms	Spec. of roles, actors and cooperative activities while using a groupware system

3.2 Process for performing a task analysis that relies on task models

Performing a task analysis requires a choice of analysis technique, and this choice influences the outcomes and insights generated from the analysis. Annett [1] and Diaper [11] both recommend the following preliminary steps when considering a task analysis: identify the scope of the analysis, identify the output format for the analysis, identify the data that needs to be collected and then select the most relevant technique.

Fig. 10 summarizes the process of task analysis, with preliminary steps on the left. Once a particular task modelling technique is selected (top middle of the Fig. 10), it is applied (right side of Fig. 10). The set of steps for the application of the chosen task modelling technique are the following: information collection and identification of user tasks, production of the task models, and validation of the task models and processing of the task models according to the objectives of the analysis.

The dotted lines around the artefacts "Most relevant task modelling technique" (middle top of Fig. 10), "Task models" (right, middle of Fig. 10) and "Application of the task modelling technique" indicate that the chosen technique and notation may not support description of all of the necessary information. The Fig. 10 therefore highlights the limitation of task modelling notations and tools for adequately fulfilling the task analysis processes – the "most relevant task analysis technique" may not fully cover the types of information that need to be represented.

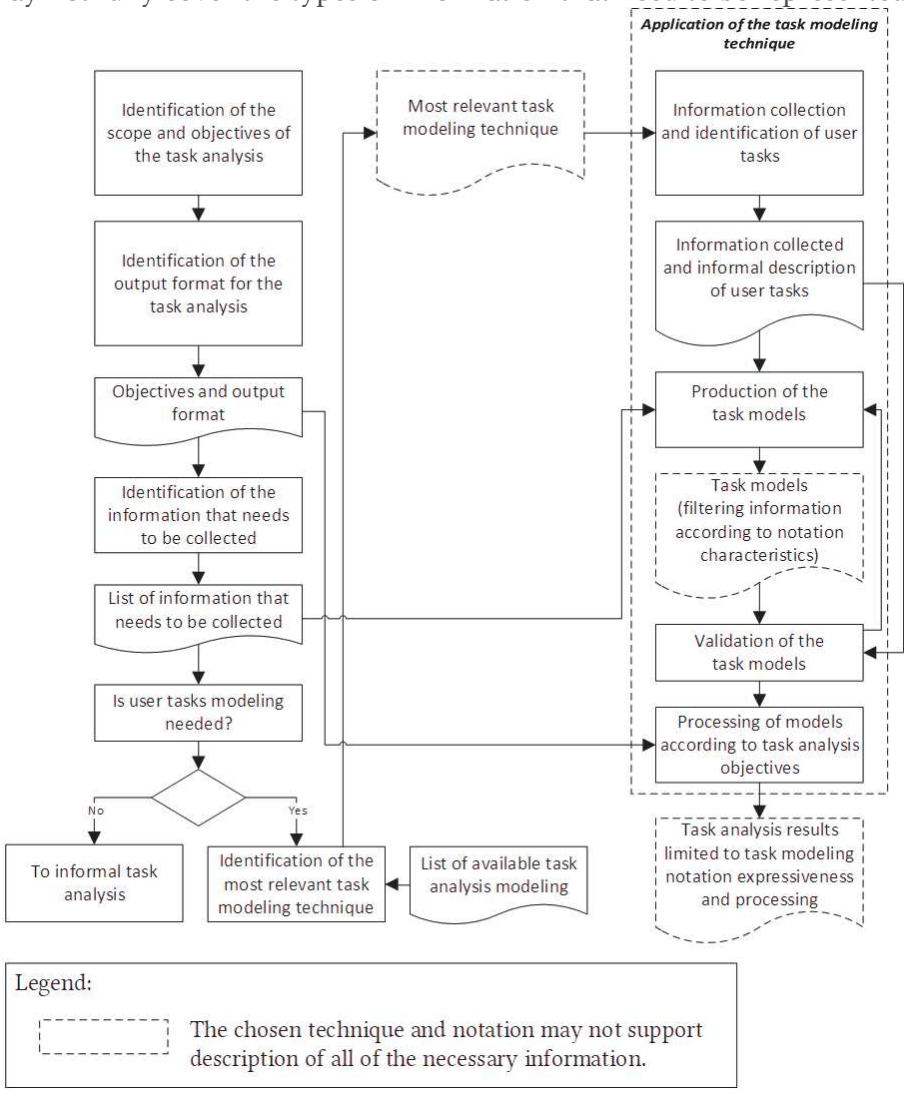

Fig. 10. Process for performing a task analysis that relies on task models

4 A PROCESS FOR CUSTOMIZING A TASK MODELLING NOTATION

Section 2 established that the choice of task analysis notation influences the resultant design insights emerging from the task analysis. It also established that when new interaction contexts emerge (such as new task domains or technologies) it is likely that gulfs of expressivity will be created due to the inability of the notation to adequately capture salient task details. In order to overcome the limitation of the task analysis processes that rely on task modelling notations, we propose a modification of the task analysis process summarized in Fig. 10. The new steps aim at enabling the customization of the task modelling notation, in order to fill the gap between its expressive power and the actual work context. We demonstrate the feasibility of the process using the example of a task modelling notation, named HAMSTERS-XL.

4.1 Process for performing a task analysis that relies on task models

Fig. 11 depicts the proposed process for customizing a task modelling notation, using a modified version of Fig. 10.

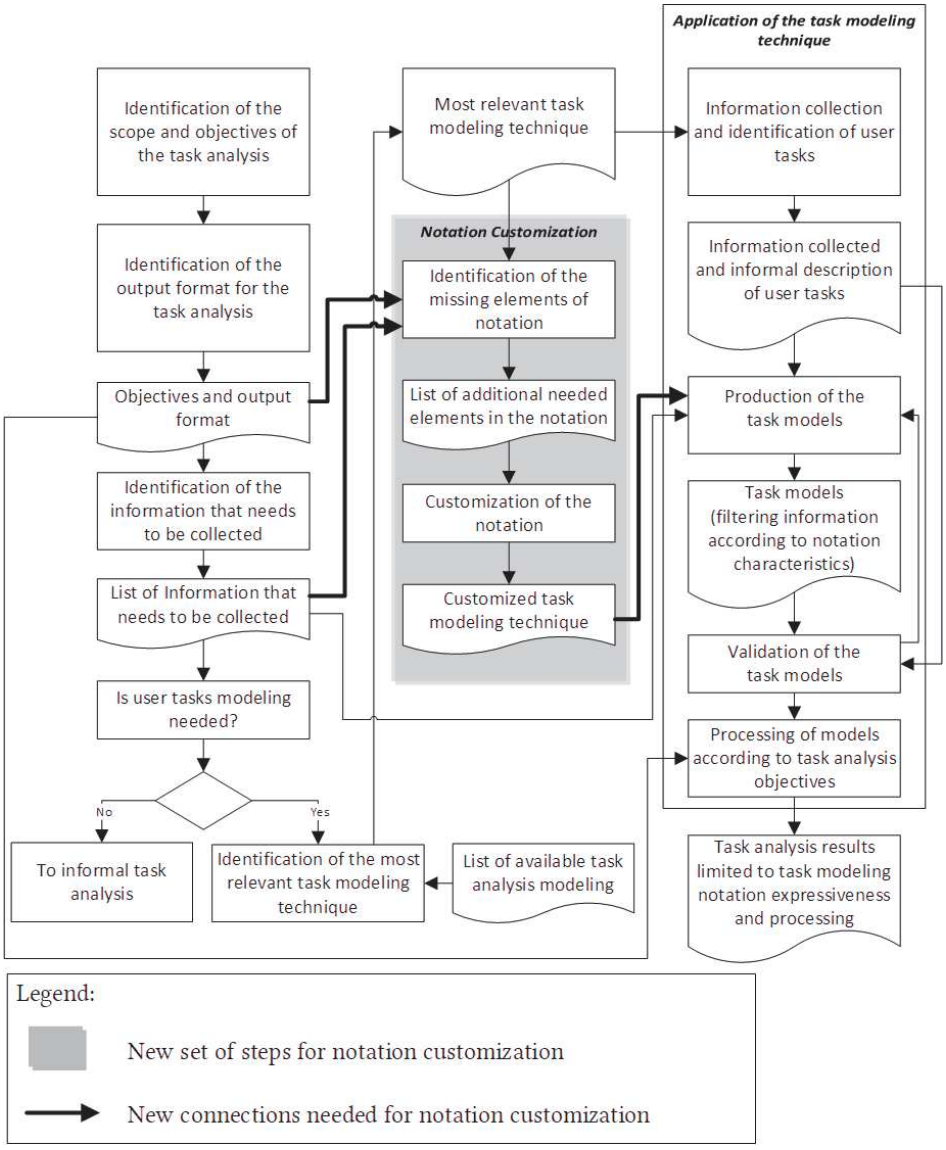

Fig. 11. Proposed process for customizing a task modelling notation for task analysis

The main additions in the process concern the notation customization steps in the middle column of the Fig. 11, which enables all of the required information to be captured by the notation (and hence the dotted elements of Fig. 10 are solid in Fig. 11).

The next section provides an example of this process using the HAMSTERS-XL notation.

4.2 The core elements of the HAMSTERS-XL notation

The HAMSTERS-XL notation is a new version of the HAMSTERS notation [40]. HAMSTERS (Human – centered Assessment and Modelling to Support Task Engineering for Resilient Systems) is a tool-supported task modelling notation for representing human activities in a hierarchical and structured way, with the intention of supporting modelling consistency, coherence and conformity between user tasks and interactive systems [34].

4.2.1 Hierarchy, task types and temporal ordering. The HAMSTERS-XL notation specifies a tree of nodes that can be tasks or temporal operators. The top node represents the main goal of the user, with lower levels representing sub-goals, tasks and actions, similar to HTA representation [1]. Fig. 2 and Fig. 8 present examples of this hierarchical decomposition.

Fig. 12 shows the palette of selectable elements in the HAMSTERS-XL notation. The parts numbered 1, 2, 3, 5 presents the main task types: abstract, user, interactive and system tasks. User task types can be refined into perceptive, motor, cognitive analysis, and cognitive decision tasks. These task types provide support for assessing automation, and, in particular, for assessing allocation of tasks and functions between the user and the system [35], as well as for comparing cognitive load between different designs (as illustrated in section 2.2).

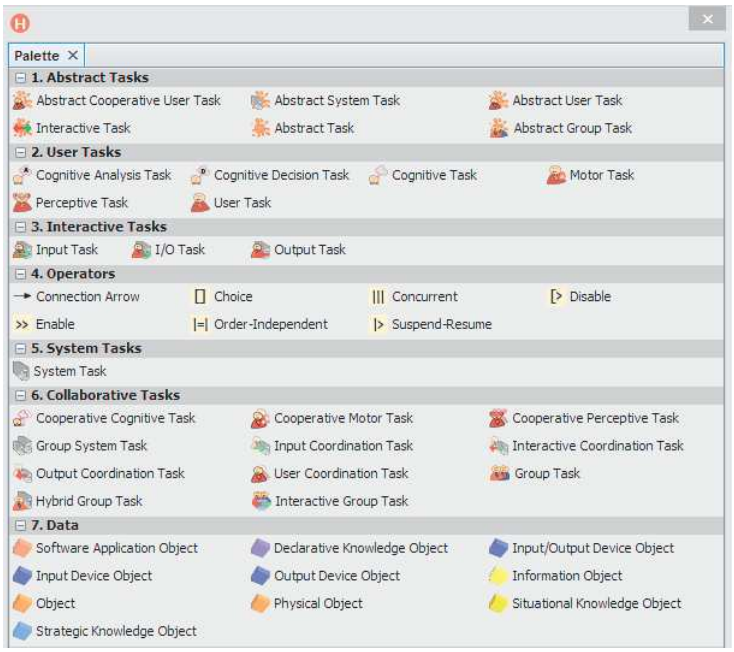

Fig. 12. Palette of the elements of notation of HAMSTERS

Temporal operators (numbered 4 in Fig. 12) are used to represent temporal relationships between sub-goals and between activities, as introduced in [55] (see [56] for the last version of the notation). In Fig. 9, the task models describe the possible ordering of user actions using the temporal operators sequence ">>" and concurrence "|||". Additional temporal characteristics can also be specified for optional, iterative, and timed tasks (illustrated in Fig. 13).

Fig. 13. Representation of temporal properties of tasks

4.2.2 Data manipulated during the accomplishment of a task. Elements of notations for representing data manipulated by the user or the system are present in several notations such as objects in CTT [55] and K-MAD [6], and knowledge in TKS [25]. HAMSTERS-XL also provides support for the explicit representation of several types of data manipulated and required for the accomplishment of a task (depicted in Fig. 14).

Fig. 14. Data manipulated during the accomplishment of a task

Fig. 14a. presents the information and knowledge data types that the user may manipulate. "Inf" nodes represent the information that the user requires, such as the "Inf: Amount needed" nodes in the ATM example shown in Fig. 8 and Fig. 9. The user may also derive information requirements from information presented by the system, such as determining the Amount Needed based on system feedback (e.g., in Fig. 8 and in Fig. 9 the perceptive task "Perceive available amount" influences the determined amount needed). The "knowledge" nodes in Fig. 14a refer to declarative, strategic and situational knowledge (the user's procedural knowledge is implicitly encoded in the structure of the task model itself).

Fig. 14b. presents the following data types: physical object, time and event. Performing a task may require to use, produce, modify, capture or release a physical object ("Phy O" followed by a text description in Fig. 14c)). For example, using an ATM requires the physical object "Card". The data type "Event" provides support for the description of events that may trigger the performance of a task. The data type "Event" provides support for describing the task triggers named "Temporal", "External event" and "Environmental cue" that have been enumerated and studied in [12].

Fig. 14c. presents data types that belong to the system side: (software) object ("Obj:" followed by a text description), software application ("SwA:" followed by a text description), input device ("in D:" followed by a text description), output device ("out D:" followed by a text description) and input/output device ("i/o D:" followed by a text description). Fig. 8 and Fig. 9 exemplify their use in describing input and output devices (keyboard, display, tactile display) as well as the objects produced by an input task (e.g., "O: requested amount").

The HAMSTERS-XL notation provides support for the description of pre-conditions for accomplishing a task (graphical elements arrows drawn between tasks and data). Pre-conditions can be simple (conditional on one type of data) or composed (conditional on several types of data). With HAMSTERS, it is possible to describe pre-conditions by drawing a "Test" arrow between the data and the task. More complex pre-conditions can be specified using Boolean expressions.

4.2.3 Additional constructs available in HAMSTERS. Complexity in models is a recurrent problem, and the time required to produce them can disincentivise their use in design projects. HAMSTERS provides three mechanisms (Sub-models, Sub-routines [40] and Components [15]) to help reduce the complexity (through structuring and reuse) and time demands of producing task models, as shown in [15] [40].

Collaborative tasks. HAMSTERS-XL supports specification of tasks that are completed individually, collaboratively and cooperatively. Collaborative work can be described at different abstraction levels: at the group level and at the individual level. A group task is a set of task that a group has to carry out in order to achieve a common goal [41], whereas a cooperative task is an individual task performed by a person in order to contribute to the achievement of the common goal [60]. Collaborative tasks may be performed with time constraints (local/distant, synchronous/asynchronous) and properties (production, communication, coordination) [33].

Human errors. HAMSTERS-XL provides notational elements that identify and describe possible human errors [14], which provides support to determine opportunities for re-design [37].

A detailed description of the main elements of the notation is available in [36].

5 HAMSTERS-XLE: HAMSTERS-XL Environment

This section presents HAMSTERS-XLE, which is the computer aided software environment for customizing, editing and simulating HAMSTERS-XL task models. It provides support for:

- editing and simulation of task models created with the HAMSTERS-XL notation,
- creating customized versions of the HAMSTERS-XL notation (adding or removal of elements of notation)

Fig. 15 shows a screenshot of HAMSTERS-XLE with the task models for the comparison of the two version of the ATM open (the central pane shows the description for the task "Withdraw Money" using the tactile version of the ATM.

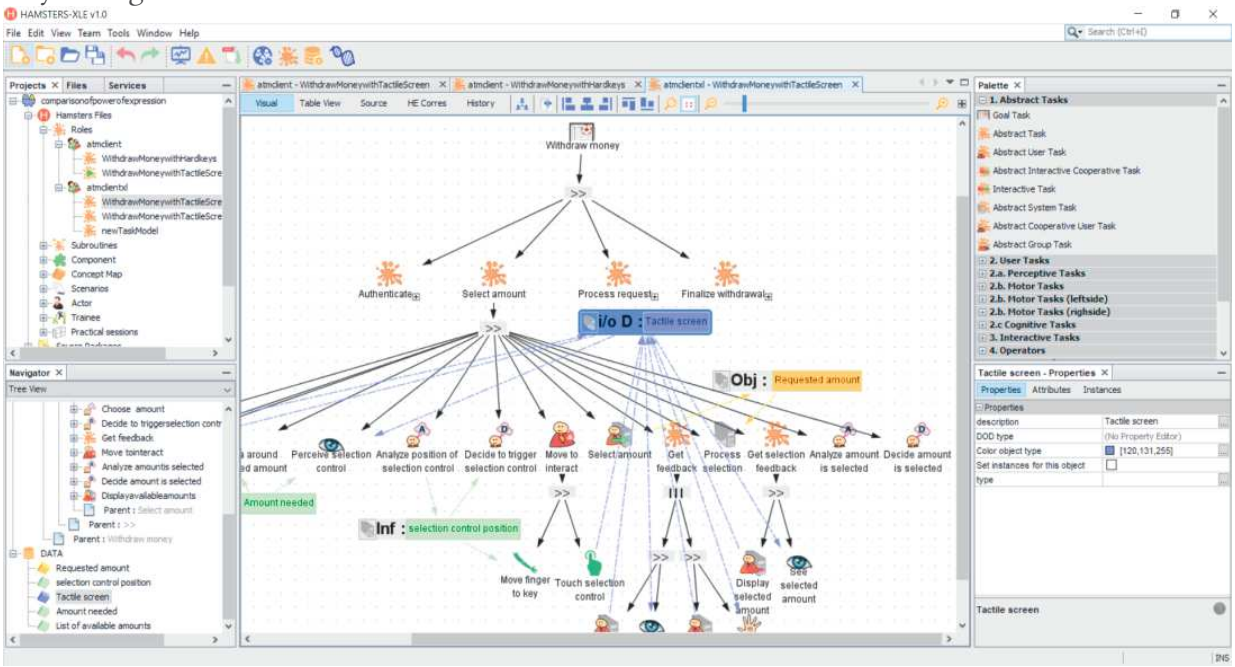

Fig. 15. Screenshot of the HAMSTERS-XL Environment

The HAMSTERS-XLE uses the Netbeans Platform [44] and Maven framework [3]. It encompasses basic IDE functions such as project management, unit testing, and versioning. As explained in Section 4.3, the elements of notation that require customization are the types of tasks and data; HAMSTERS-XLE supports customization of these elements, using the steps shown in Fig. 16.

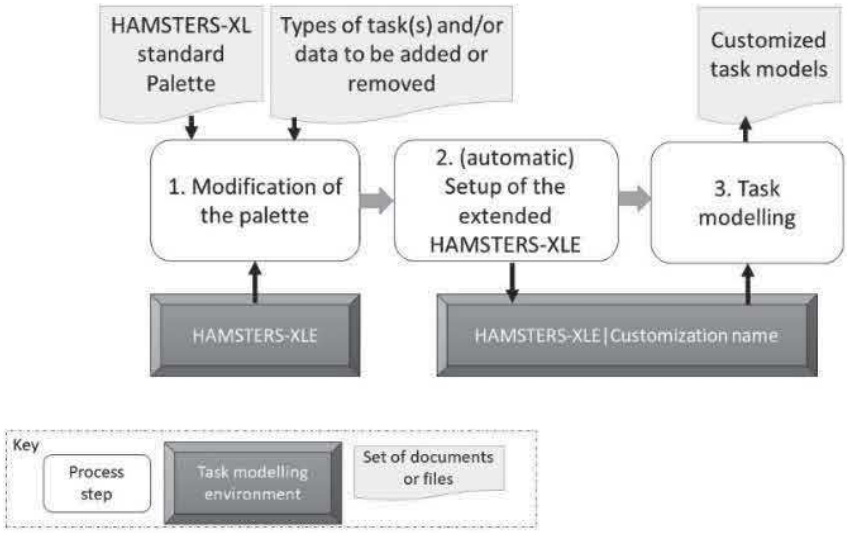

Fig. 16. Steps for using HAMSTERS-XLE for customizing HAMSTERS-XL notation

6 VALIDATION OF THE PROCESS FROM EXAMPLES

To demonstrate the feasibility and usefulness of the customization process, this section presents small examples from case studies of task analysis.

6.1 HAMSTERS-XL|Tactile ATM

This first example presents customization of the HAMSTERS-XL notation in support of the Tactile ATM example introduced in Section 2.2. Five customizations were required to cover specific task elements, as summarized in see Table 4.

Table 4. Proposed customizations for HAMSTERS-XL|Tactile ATM

Icon					
Description	Move arm motoric task	Finger press motoric task	Finger touch (put on) motoric task	Sight perceptive task	Hear perceptive task

Fig. 17 presents the task model for the goal “Withdraw money” using an ATM with hard keys and an output only screen. The sequence of high-level tasks to reach the goal are: to authenticate, to select amount, the ATM has to process request and, to finalize withdraw by taking the money and card. The Fig. 17 details “Select amount” abstract task with the following sequence. First, the ATM displays available amounts and the user perceives them and choose one among them. Then, the user has to reach the associated hard key (“Move arm to key” Right arm move motoric task and “Put finger on the button” Right hand finger put on motoric task) and to press it to communicate her/his choice to the ATM (“Press key” Right hand finger press motoric task and “Select amount” Interactive input task). The user perceives the sound feedback (“Ear sound beep” Ear perceptive task) and the selected amount. Finally, he/she analyses and determines that she/he correctly selected the desired amount.

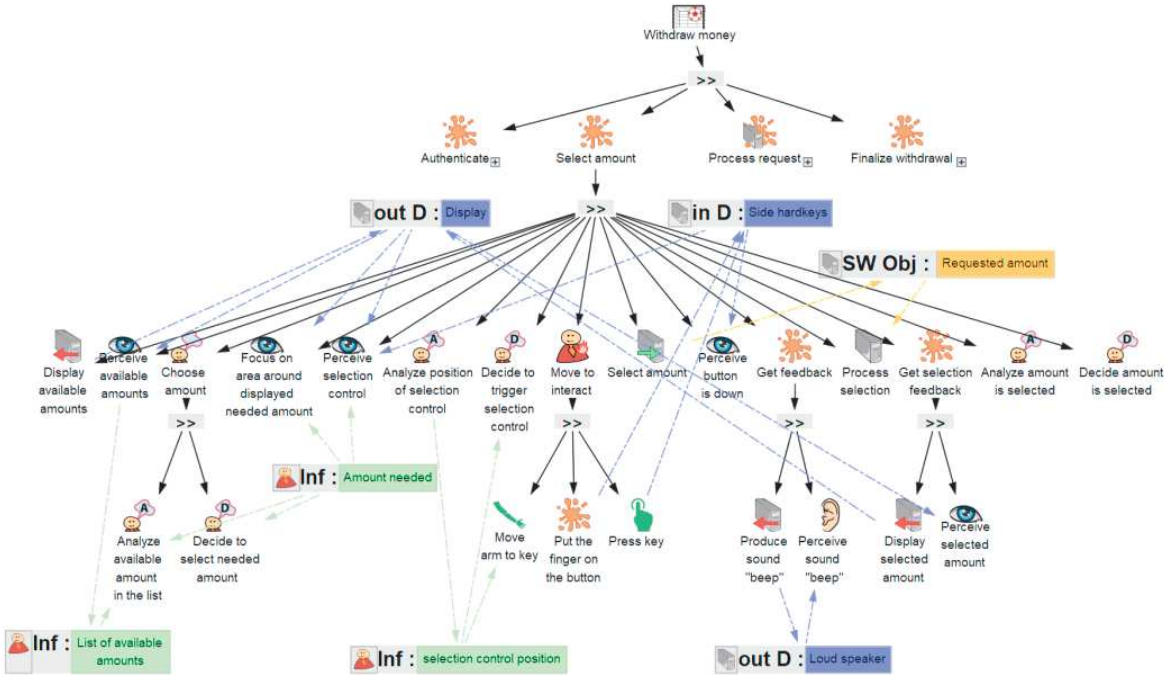

Fig. 17. Task model of the user task "Withdraw money" with the "hard keys + output only screen" version of the ATM produced using the customization HAMSTERS-XL|Tactile ATM

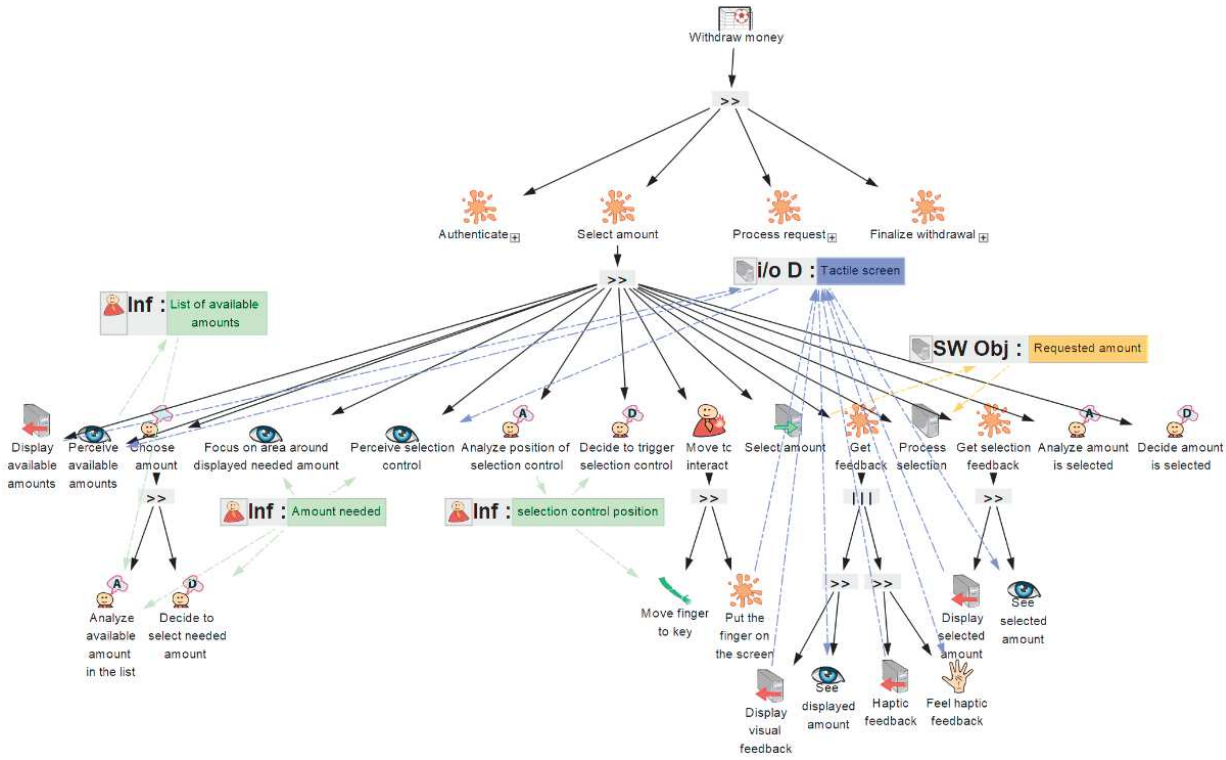

Fig. 18. Task model of the user task "Withdraw money" with the "fully tactile screen" version of the ATM produced using the customization HAMSTERS-XL|Tactile ATM

Fig. 18 presents the task model for the user goal "Withdraw money" using an ATM with a fully tactile screen. The tasks to reach the goal are the same as the previous described model except for the motoric task and feedbacks tasks. The user just has to put the finger on the screen to trigger the interaction ("Put the finger on the screen" finger touch motoric task) whereas the user has to

press the button in the previous model. Moreover, there is no sound feedback with this version of ATM but a haptic feedback perceived by the user ("Feel haptic feedback" touch perceptive task).

Table 5 provides a summary comparison of the original HAMSTERS models and the HAMSTERS-XL models for the hardkey and tactile ATMs. The customized HAMSTERS-XL models provide additional insights into specific perceptive and motoric actions that were unavailable with the original models.

Table 5. Comparison of the number of tasks per type for the two versions of the ATM

	HAMSTERS models for "hardkeys + output only display" ATM	HAMSTERS models for "fully tactile display" ATM	HAMSTERS XL models for "hardkeys + output only display" ATM	HAMSTERS XL models for "fully tactile display" ATM
User	15	14	16	15
Perceptive	6	5	6	6
See	NA	NA	5	5
Hear	NA	NA	1	0
Feel			0	1
Cognitive	6	6	6	6
Cognitive analysis	3	3	3	3
Cognitive decision	3	3	3	3
Motoric	2	2	3	2
Move arm	NA	NA	1	1
Press	NA	NA	1	0
Put on [button/screen]	NA	NA	1	1
Interactive	1	1	1	1
Interactive input	1	1	1	1
System	4	5	3	4
Process	1	1	1	1
Interactive output	3	4	3	4
Data	7	5	7	5
Output Device	2	0	2	0
Input Device	1	0	1	0
Input/Output Device		1		1
Object	1	1	1	1
Information	3	3	3	3

6.2 HAMSTERS-XL|Taste

Next, we describe customization in Hamsters-XL for tasks based on Vi and Obrist's experiment [63] concerning the influence of taste on risk-taking behaviour. The main user task for this experiment is the standardized Balloon Analogue Risk-Taking (BART) task, which involves inflating air balloons. The main measure of the risk-taking behaviour is function of the average value of the number of pumps for unexploded balloons. Fig. 19 describes the procedure of the experiment. First, participants answer two questionnaires. Then, they ingest a stimulus (a taste or a neutral). Finally, they pump-up a virtual balloon and decide to stop and cash out to collect the points reward (later converted in money). However, the balloon either can inflate or explode randomly. If the balloon explodes, the participant loses collected points.

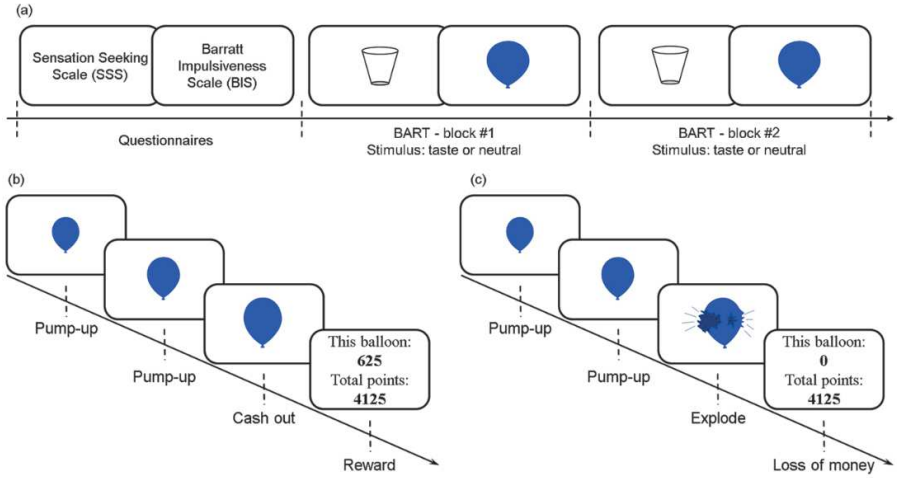

Fig. 19. Overview of the experimental procedure from [63]

The customizations required for this task are summarised in Table 6.

Table 6. Proposed customizations for HAMSTERS-XL|Taste

Icon		
Name	Taste perceptive task	Taste information (the taste perceived by the participant)

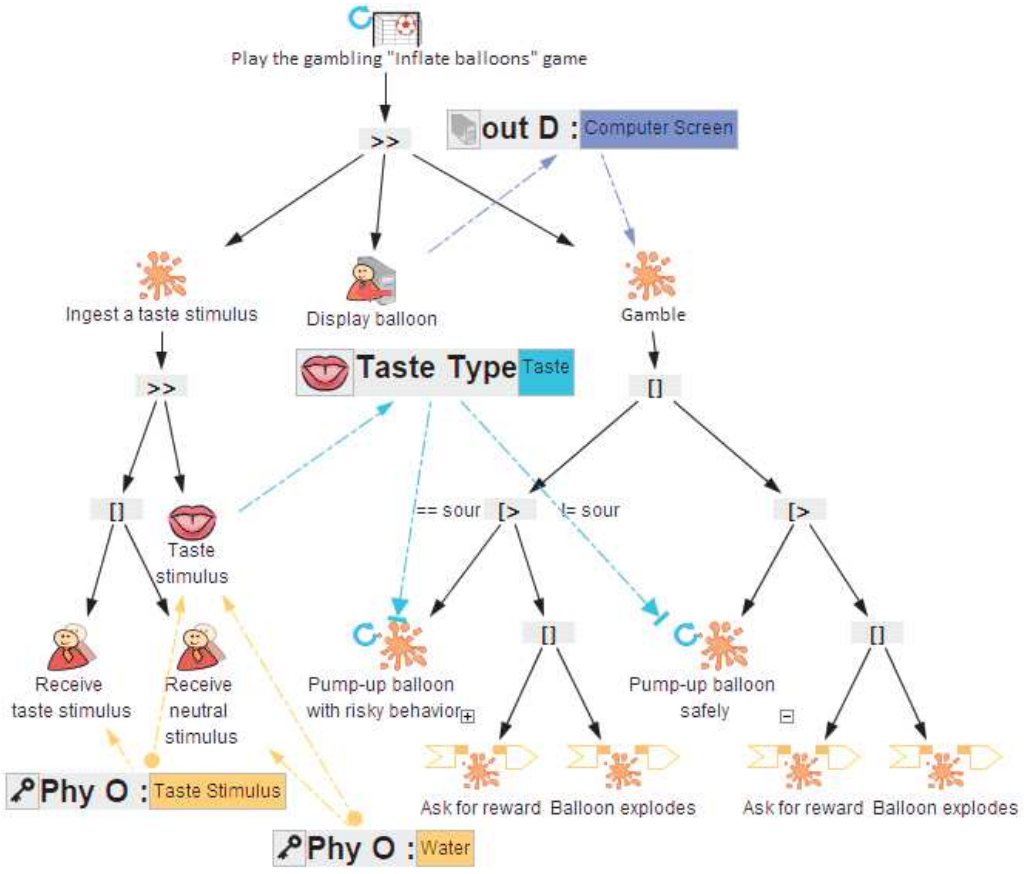

Fig. 20. Task model of the user goal “Play the gambling “Inflate balloons” game” produced using the customization HAMSTERS-XL|Taste

The Fig. 20 presents the task model “Play the gambling “inflate balloon” game”. To reach the goal the user has first to ingest a taste stimulus or a neutral taste stimulus: water (“Taste stimulus” Taste perceptive task). Then, depending on the ingested taste, the user may adopt a risky or a safe behaviour to pump-up the balloon (test on the value of “Taste” Taste Type for the “Pump-up balloon with risky behaviour” and “Pump-up balloon safely” Abstract iterative tasks).

6.3 HAMSTERS-XL|Cockpit

The final example examines the pilots’ task of modifying the display range of the Navigation Display in a commercial aircraft cockpit, with two different types of Flight Control Unit (a mechanical version with physical buttons and knobs and a graphical user interface version controlled with a keyboard and Control Cursor Unit).

In an aircraft cockpit, the navigation display (ND) is one of the key element that help pilots understanding the aircraft environment. Both pilots have their own ND in front of them. The ND is capable of displaying, amongst other data:

- The aircraft progress on the flight plan (the green line represented in Fig. 21c),
- The radar image produced by the aircraft weather radar showing density of clouds ahead of the aircraft (red/yellow/green images represented in Fig. 21c).

In order to plan for both short and long-term actions related to weather conditions, pilots frequently use the ability to modify the visualization range of the ND. The higher the range, the further the radar inspects weather conditions but the less precise the information is.

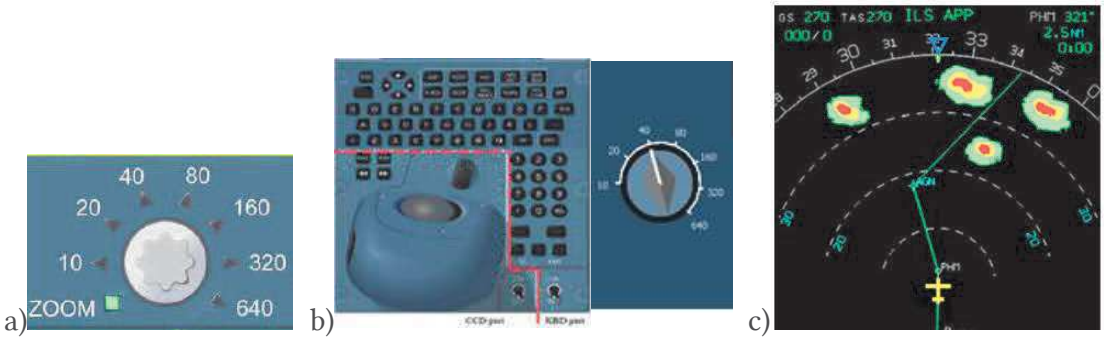

Fig. 21. a) Knob (at the left) for manipulation of visualization range of the ND; b) the KCCU (at the left) for the manipulation of the virtual knob (at the right) to update visualization range of the ND; c) the ND

Analysis of the pilots’ task shows that the customizations shown in Table 7 need to be included within HAMSTERS-XL.

Table 7. Proposed customizations for HAMSTERS-XL|Cockpit

Icon							
Description	Grip motoric task (Grip knob)	Grip and turn motoric task (Grip and turn right knob)	Finger press motoric task (Press KCCU button)	Grab motoric task (Grab KCCU),	Move fingers motoric task (Move KCCU ball)	Sight perceptive task (Locating controls, assessing aircraft status from the screens, etc.)	Touch/feel perceptive task (Identifying control shapes, perceiving feedback from controls, etc.)

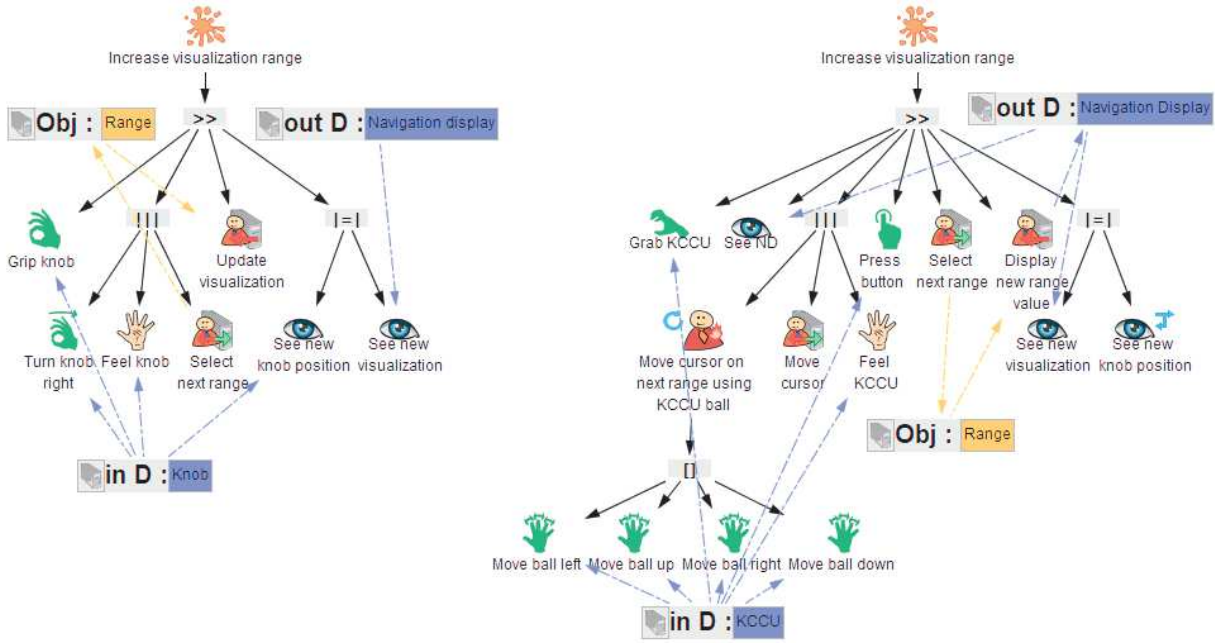

Fig. 22. Task models for the task "increase visualization range" in the aircraft cockpit (with the physical knob on the left side and with the virtual knob on the right side)

Fig. 22 shows two different task models for the user "Increase visualization range" task. The task model at the left details the task with the physical knob. First, the user has to grip the knob ("Grip knob" Right hand grip motoric task). Then, the user feels the knob against her/his hand and turns right the knob to reach the next value. The task model at the right details the task "Increase visualization range" with the KCCU and the virtual knob. First, the user has to grab the KCCU ("Grab KCCU" Right hand grab motoric task). Then, the user has to move the KCCU ball with the index, the middle and the ring fingers ("Move ball left/up/right/down" Right hand move three fingers task) to move the cursor on the next value. Finally, the user presses the KCCU button ("Press button" Right hand finger press motoric task) to select the next range value.

Table 8 provides a summary of the quantity of different types of activities required to complete tasks with the physical and virtual knob versions of the interfaces, highlighting the need for additional motor actions with the virtual knob.

Table 8. Comparison of the number of tasks per type for the two versions of the ATM

		HAMSTERS XL models for "Increase visualization range" with physical knob version	HAMSTERS XL models for "Increase visualization range" with virtual knob version
User		5	5+n
	Perceptive	3	3
	See	2	2
	Feel	1	1
	Motoric	2	2+n
	Grip	1	0
	Grip and Turn	1	0
	Move 3 fingers	0	n
	Grab	0	1
	Press	0	1
Interactive		1	1
	Input/Output	1	1
System		1	1
	Output	1	1
Manipulated data		3	3
	Object	1	1
	Output device	1	1
	Input device	1	1

6.4 Feedback from users

HAMSTERS-XLE provides support to perform the main user tasks when editing task models: « Add task to a model », « Add operators »... (see the list of main user tasks for task modelling identified by Vigo et al. [64]). Additional task modelling user needs have been gathered during several research and industrial projects, as well as during tutorial sessions at ACM CHI conferences [50] [49], INTERACT conference [51] and EUROCONTROL [13]. We had about 60 users from various application domains (e.g. web, business applications, health and medical appliances, military and civil aircrafts, air traffic management) and the main user profiles were software engineers, human factor experts and project managers (ranging from novice to senior in their job). Task type customization was the most recurring need, followed by the task properties customization. For example, UI software designers and project managers of the Air Traffic Management domain argued that a particular type of task that is very important and that is frequently performed by air traffic controllers needed to be added to the notation. They commonly name this type of task "elbow communication" (it is a nonverbal exchange of information between two controllers). They can now add this task type and perform task modelling by applying the proposed process with HAMSTERS-XLE.

Concerning the application of the process with HAMSTERS-XLE, we gathered qualitative user feedback from the engineers and researchers being part of the research and industrial projects. The illustrative example presented in section 6.4 is an excerpt of task models that have been produced during an industrial project with a commercial aircraft manufacturer. The proposed process was fully applied several times by 5 different users (engineers or researchers) with HAMSTERS-XLE and thus the effectiveness criteria of usability is reached.

HAMSTERS-XL is publicly available to the community[1] and we continuously try to gather feedback and insights from users.

[1] https://www.irit.fr/recherches/ICS/softwares/hamsters/

7 CONCLUSION

Task models provide a precise means for making explicit and analysing user goals and how the activities required to complete them. However, task modelling techniques typically do not evolve at the same pace as interactive technologies, leading to a gulf between modelling needs and the expressive capabilities of the modelling environments.

This paper has highlighted these problems, and it has proposed the use of customizable task modelling notations and associated tools to ease the problems. The main contributions are:

- demonstration of the need for customizable task modelling techniques in order to be able to produce task models that fit the scope and objectives of the task analysis,
- development of a process and principles for identifying the customization needs and an instantiation of this process with the HAMSTERS-XL task modelling technique,
- demonstration of the feasibility of the approach with a dedicated tool (HAMSTERS-XLE) supporting the customization process and the exploitation of the customized notation (editing and simulating of task models),
- the application of several customizations to a set of illustrative examples extracted from larger case studies, demonstrating the applicability and the benefits of customizing task modelling techniques.

The tool supported notation is publicly available (see additional files in the submission).

It is important to note that the customization of notations comes with an increase of the number of elements of a notation and of the number of elements in the task models (describing the work of the users with that notation). This increase adds costs and time to the modelling activities and has to be carefully assessed by the analysts. However, this trade-off is the same as when using any modelling technique and processes (be it in task analysis domain or in the software engineering one, for instance). It is beyond the scope of this paper to argue about such tradeoffs and whether modelling is a good strategy or not. However, clearly, specific application domains (such as safety critical ones [43]) benefit from precise understanding of tasks and associated issues such as operators' workload.

The need for customized modelling notations is not specific to task modelling notation but has been proved of interest for most design activities of interactive systems (user interface layout [27] [29], system behaviour [21] [52], system architecture [9] [22]…). In computer science domain and in software engineering in particular, some of the modelling tools thus provide means for customization (of course very different from describing user behaviour) and also tries to allow users tuning the tool so that they can fit better to their needs and their work. For example, Langer et al. proposed an Eclipse plugin to support extensibility of Domain Specific Modelling Languages [28]. An interesting topic for future work would be generalize the work done here, to identify abstractions and to use that abstract model to compare customization needs for task analysis and for software engineering. Even though very complex and requiring the analysis of a lot of tools and notations, such work might identify conceptual concepts to support customization in a generic way as this has been done in the area of end user customization of application [4] [30].

REFERENCES

[1] John Anett. 2004. Hierarchical Task Analysis. In Diaper Dan, Stanton Neville (Eds), *The Handbook of Task Analysis for Human-Computer Interaction* (pp. 67-82). Lawrence Erlbaum Associates.

[2] John Annett, Keith Duncan.1967. Task analysis and training design. *Occupational psychology*, 41, 211-221.

[3] Apache Maven Project, https://maven.apache.org/index.html, last access March 2018.

[4] Nikola Banovic, Fanny Chevalier, Tovi Grossman, and George Fitzmaurice. 2012. Triggering triggers and burying barriers to customizing software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2717-2726.

[5] Gregor Buchholz, Peter Forbrig. 2017. Extended Features of Task Models for Specifying Cooperative Activities. In *Proc. of ACM on EICS*, 1, 7:1-7:21.

[6] Sybille Caffiau, Dominique Scapin, Patrick Girard, Mickaël Baron, and Francis Jambon. 2010. Increasing the expressive power of task analysis: Systematic comparison and empirical assessment of tool-supported task models. *Interact. with Comput.* 22, 6 (November 2010), 569-593.

[7] Cockton, G., & Woolrych, A. (2001). Understanding inspection methods: Lessons from an assessment of heuristic evaluation. *People and Computers XV, joint Proceedings of HCI 2001 and IHM 2001*, Springer Verlag 171–192.
[8] José Creissac Campos, Camille Fayollas, Marcelo Gonçalves, Célia Martinie, David Navarre, Philippe Palanque, and Miguel Pinto. 2017. A More Intelligent Test Case Generation Approach through Task Models Manipulation. Proc. ACM Hum.-Comput. Interact. 1, EICS, Article 9 (June 2017), 20 pages.
[9] Martin Cronel, Bruno Dumas, Philippe A. Palanque, Alexandre Canny. 2018. MIODMIT: A Generic Architecture for Dynamic Multimodal Interactive Systems. In Proc. of *IFIP TC13.2 Conference on Human Centered Software Engineering*, HCSE 2018, 109-129.
[10] Dan Diaper. 1990. Task Analysis for Knowledge Descriptions (TAKD): The Method and an Example. In *Task Analysis for Human-Computer Interaction*, D. Diaper (ed.), Ellis Horwood, pp. 108-159.
[11] Dan Diaper. 2004. Understanding Task Analysis. In Diaper Dan, Stanton Neville (Eds), *The Handbook of Task Analysis for Human-Computer Interaction* (pp. 67-82). Lawrence Erlbaum Associates.
[12] Alan Dix, D. Ramduny-Ellis, J. Wilkinson. 2004. Trigger Analysis - understanding broken tasks. Chapter 19 in *The Handbook of Task Analysis for Human-Computer Interaction*. D. Diaper & N. Stanton (eds.). Lawrence Erlbaum Associates, 2004. pp. 381-400.
[13] Fabrice Drogoul, Philippe Palanque. Design and Assessment of Systems Using Human Centered Approaches. Eurocontrol training. https://trainingzone.eurocontrol.int/ilp/pages/coursedescription.jsf?courseId=6453924&visibleMetadatas=registration-identifier,course-content-learning-form,skills,location,free-places,idd,price,languages&catalogId=896431&templateId=6812197&employeeId=-1&programmeId=-1, last accessed Feb. 2019.
[14] Racim Fahssi, Celia Martinie, Philippe Palanque. 2015. Enhanced Task Modelling for Systematic Identification and Explicit Representation of Human Errors. In: Abascal J., Barbosa S., Fetter M., Gross T., Palanque P., Winckler M. (eds) *IFIP TC13 Conference on Human-Computer Interaction – INTERACT 2015*. Lecture Notes in Computer Science, vol 9299.
[15] Peter Forbrig, Célia Martinie, Philippe Palanque, Marco Winckler, and Racim Fahssi. 2014. Rapid Task-Models Development Using Sub-models, Sub-routines and Generic Components. In *Proceedings of the 5th IFIP WG 13.2 International Conference on Human-Centered Software Engineering* - Volume 8742 (HCSE 2014), Stefan Sauer, Cristian Bogdan, Peter Forbrig, Regina Bernhaupt, and Marco Winckler (Eds.), Vol. 8742. Springer-Verlag New York, Inc., New York, NY, USA, 144-163.
[16] Werner Gaulke and Jürgen Ziegler. 2016. Rule-enhanced task models for increased expressiveness and compactness. In *Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '16)*. ACM, New York, NY, USA, 4-15.
[17] Matthias Giese, Tomasz Mistrzyk, Andreas Pfau, Gerd Szwillus, and Michael Detten. 2008. AMBOSS: A Task Modelling Approach for Safety-Critical Systems. In *Proceedings of the 2nd Conference on Human-Centered Software Engineering and 7th International Workshop on Task Models and Diagrams (HCSE-TAMODIA '08)*, Peter Forbrig and Fabio Paternò (Eds.). Springer-Verlag, Berlin, Heidelberg, 98-109.
[18] Gong, R. & Elkerton, J. (1990). Designing minimal documentation using the GOMS model: A usability evaluation of an engineering approach. *CHI 90 Proceedings*. New York, ACM DL.
[19] Saul Greenberg. Working through Task-Centered System Design. In Diaper, D. and Stanton, N. (Eds) *The Handbook of Task Analysis for Human-Computer Interaction*. Lawrence Erlbaum Associates (2004). p49-66.
[20] Valeria Gribova. A method of Context-Sensitive Help Generation Using a Task Project. *International Journal on Information Theories & Applications* Vol.15, pp. 391-395, 2008.
[21] Arnaud Hamon, Philippe Palanque, José Luís Silva, Yannick Deleris, and Eric Barboni. 2013. Formal description of multi-touch interactions. In *Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems (EICS '13)*. ACM, New York, NY, USA, 207-216.
[22] Lode Hoste, Bruno Dumas, and Beat Signer. 2011. Mudra: a unified multimodal interaction framework. In *Proceedings of the 13th international conference on multimodal interfaces (ICMI '11)*. ACM, New York, NY, USA, 97-104.
[23] Bonnie E. John and David E. Kieras. 1996. The GOMS family of user interface analysis techniques: comparison and contrast. *ACM Trans. Comput.-Hum. Interact.* 3, 4 (December 1996), 320-351.
[24] Peter Johnson. 1992. Human-Computer Interaction: psychology, task analysis and software engineering, McGraw Hill, Maidenhead, UK.
[25] Peter Johnson, H. Johnson and F. Hamilton. 2000. Getting the Knowledge into HCI: Theoretical and Practical Aspects of Task Knowledge Structures. In. *Cognitive Task Analysis*. J. Schraagen, S. Chipman, V. Shalin LEA
[26] Frédéric Jourde, Yann Laurillau, and Laurence Nigay. 2010. COMM notation for specifying collaborative and multimodal interactive systems. In *Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '10)*. ACM, New York, NY, USA, 125-134.
[27] James A. Landay. 1996. SILK: sketching interfaces like krazy. In *Conference Companion on Human Factors in Computing Systems (CHI '96)*, Michael J. Tauber (Ed.). ACM, New York, NY, USA, 398-399.
[28] Philip Langer, Konrad Wieland, Manuel Wimmer, Jordi Cabot. (2011) From UML Profiles to EMF Profiles and Beyond. In: Bishop J., Vallecillo A. (eds) Objects, Models, Components, Patterns. *TOOLS 2011*. Lecture Notes in Computer Science, vol 6705. Springer, Berlin, Heidelberg.

[29] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. 2000. DENIM: finding a tighter fit between tools and practice for Web site design. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '00)*. ACM, New York, NY, USA, 510-517.
[30] Wendy Mackay. 1991. Triggers and barriers to customizing software. In *Proc. CHI '91*. ACM, 153-160.
[31] Marco Manca , Fabio Paternò, Carmen Santoro. 2016. Collaborative Task Modelling on the Web. In: Bogdan C. et al. (eds) Human-Centered and Error-Resilient Systems Development. *HESSD 2016, HCSE 2016*. Lecture Notes in Computer Science, vol 9856. Springer, Cham.
[32] Marco Manca, Fabio Paterno, Carmen Santoro, Lucio Davide Spano. 2014. Considering task pre-conditions in model-based user interface design and generation. In *Proc. of the EICS '14*, 149-154, ACM.
[33] Célia Martinie, Eric Barboni, David Navarre, Philippe Palanque, Racim Fahssi, Erwann Poupart, and Eliane Cubero-Castan. 2014. Multi-models-based engineering of collaborative systems: application to collision avoidance operations for spacecraft. In *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems (EICS '14)*. ACM, New York, NY, USA, 85-94.
[34] Célia Martinie, David Navarre, Philippe Palanque, and Camille Fayollas. 2015. A generic tool-supported framework for coupling task models and interactive applications. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '15)*. ACM, New York, NY, USA, 244-253.
[35] Célia Martinie, Philippe Palanque, Eric Barboni and Martina Ragosta. 2011. Task-model based assessment of automation levels: Application to space ground segments. *IEEE International Conference on Systems, Man, and Cybernetics*, Anchorage, pp. 3267-3273.
[36] Célia Martinie, Philippe Palanque, Elodie Bouzekri, Eric Barboni, Alexandre Canny. Principles of Task Analysis and Modelling: Understanding activity Modeling tasks and Analysing models. In Handbook of Human Computer Interaction, Jean Vanderdonckt (Ed.), Springer, to appear, https://www.springer.com/us/book/9783319732282
[37] Célia Martinie, Philippe A. Palanque, Racim Fahssi, Jean-Paul Blanquart, Camille Fayollas, Christel Seguin. 2016. Task Model-Based Systematic Analysis of Both System Failures and Human Errors. IEEE Trans. Human-Machine Systems 46(2), 243-254.
[38] Célia Martinie, Philippe Palanque, David Navarre, Marco Winckler, and Erwann Poupart. 2011. Model-based training: an approach supporting operability of critical interactive systems. In *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '11)*. ACM, New York, NY, USA, 53-62.
[39] Célia Martinie, Philippe Palanque, Martina Ragosta, and Racim Fahssi. 2013. Extending procedural task models by systematic explicit integration of objects, knowledge and information. In *Proceedings of the 31st European Conference on Cognitive Ergonomics (ECCE '13)*. ACM, New York, NY, USA, Article 23, 10 pages.
[40] Célia Martinie, Philippe Palanque, and Marco Winckler. 2011. Structuring and composition mechanisms to address scalability issues in task models. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part III (INTERACT'11)*, Pedro Campos, Nuno Nunes, Nicholas Graham, Joaquim Jorge, and Philippe Palanque (Eds.), Vol. Part III. Springer-Verlag, Berlin, Heidelberg, 589-609.
[41] J.E. McGrath. 1984. *Groups: Interaction and Performance*. Prentice Hall, Inc., Englewood Cliffs.
[42] Giulio Mori, Fabio Paternò, and Carmen Santoro. 2002. CTTE: support for developing and analyzing task models for interactive system design. *IEEE Trans. Softw. Eng.* 28, 8 (August 2002), 797-813.
[43] David Navarre, Philippe Palanque, and Sandra Basnyat. 2008. A Formal Approach for User Interaction Reconfiguration of Safety Critical Interactive Systems. In Proceedings of the 27th international conference on Computer Safety, Reliability, and Security (SAFECOMP '08), Michael D. Harrison and Mark-Alexander Sujan (Eds.). Springer-Verlag, Berlin, Heidelberg, 373-386.
[44] Netbeans Platform, https://netbeans.org/features/platform/index.html, last accessed September 2018.
[45] O'Donnell, R. D.; Eggemeier, F. T. Workload Assessment Methodology; In K. R. Boff & L. Kaufman & J. P. Thomas (Eds.), *Handbook of Perception and Human Performance* (Vol. II Cognitive Processes and Performance, pp. 42-41 - 42-49). Wiley & Sons, 1986.
[46] Eamonn O'Neill and Peter Johnson. 2004. Participatory task modelling: users and developers modelling users' tasks and domains. In *Proceedings of the 3rd annual conference on Task models and diagrams (TAMODIA '04)*. ACM, New York, NY, USA, 67-74.
[47] Philippe Palanque, Rémi Bastide, Louis Dourte. Contextual Help for Free with Formal Dialogue Design. In *Proc. of HCI International* 1993.
[48] Philippe Palanque, Célia Martinie, and Camille Fayollas. 2018. Automation: Danger or Opportunity? Designing and Assessing Automation for Interactive Systems. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18)*. ACM, New York, NY, USA, Paper C19, 4 pages.
[49] Philippe Palanque and Célia Martinie. 2016. Designing and Assessing Interactive Systems Using Task Models. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. ACM, New York, NY, USA, 976-979.
[50] Philippe Palanque and Célia Martinie. 2015. Designing and Assessing Interactive Systems Using Task Models. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 2465-2466.
[51] Philippe Palanque, Célia Martinie, and Marco Winckler. 2017. Designing and Assessing Interactive Systems Using Task Models. In *16th IFIP TC 13 International Conference on Human-Computer Interaction --- INTERACT 2017* - Volume 10516, Regina Bernhaupt, Girish Dalvi, Anirudha Joshi, Devanuj K. Balkrishan, Jacki O'Neill, and Marco Winckler (Eds.), Vol. 10516. Springer-Verlag, Berlin, Heidelberg, 383-386

[52] Philippe A. Palanque, Amélie Schyn. 2003. A Model-Based Approach for Engineering Multimodal Interactive Systems. *IFIP TC 13 Conference on Human Computer Interaction, INTERACT 2003*.
[53] Pangoli S., Paternò F. Automatic Generation of Task-Oriented Help. *ACM Symposium on UIST 1995*, 181-187.
[54] Fabio Paterno. Task models in interactive software systems, *Handbook of Software Engineering and Knowledge Engineering*, Vol 1, 2002, Publisher: World Scientific, pp. 1-19.
[55] Fabio Paternò, Cristiano Mancini, Silvia Meniconi. ConcurTaskTrees. 1997. A Diagrammatic Notation for Specifying Task Models. In *Proc. of IFIP INTERACT 1997*, pp. 362-369.
[56] Fabio Paterno, Carmen Santoro, Lucio Davide Spano. 2012. Concur Tast Trees (CTT), W3C Working Group Submission, www.w3.org/2012/02/ctt/, last accessed March 2018.
[57] Fabio Paternò and Enrico Zini. 2004. Applying information visualization techniques to visual representations of task models. In *Proceedings of the 3rd annual conference on Task models and diagrams (TAMODIA '04)*. ACM, New York, NY, USA, 105-111.
[58] David Pinelle, Carl Gutwin, and Saul Greenberg. 2003. Task analysis for groupware usability evaluation: Modelling shared-workspace tasks with the mechanics of collaboration. *ACM Trans. Comput.-Hum. Interact.* 10, 4 (December 2003), 281-311.
[59] Martina Ragosta, Célia Martinie, Philippe Palanque, David Navarre, and Mark Alexander Sujan. 2015. Concept Maps for Integrating Modelling Techniques for the Analysis and Re-Design of Partly-Autonomous Interactive Systems. In *Proc. of the 5th International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS '15)*, ACM, New York, NY, USA, 41-52.
[60] J. Roschelle & S.D. Teasley. 1995. The construction of shared knowledge in collaborative problem solving. In C. E. O'Malley (Ed.), *Computer-supported collaborative learning* (pp. 69-197).
[61] Daniel Sinnig, Maik Wurdel, Peter Forbrig, Patrice Chalin, Ferhat Khendek. Practical Extensions for Task Models. *In proc. of TAMODIA 2007*, 42-55, Springer.
[62] Gerrit C. van der Veer, Bert F. Lenting, Bas A.J. Bergevoet. 1996. GTA: Groupware task analysis — Modelling complexity, *Acta Psychologica*, Volume 91, Issue 3, pages 297-322.
[63] Chi Thanh Vi and Marianna Obrist. 2018. Sour Promotes Risk-Taking: An Investigation into the Effect of Taste on Risk-Taking Behaviour in Humans. *Scientific Reports* 8, 1: 7987.
[64] Markel Vigo, Carmen Santoro and Fabio Paternò. 2017. The usability of task modelling tools. *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Raleigh, NC, 2017, pp. 95-99.
[65] Marco Winckler, Philippe Palanque, and Carla M. D. S. Freitas. 2004. Tasks and scenario-based evaluation of information visualization techniques. In Proceedings of the 3rd annual conference on Task models and diagrams (TAMODIA '04). ACM, New York, NY, USA, 165-172.